LA-UR-02-1224
Approved for public release; distribution is unlimited.

Title: **Fast, efficient error reconciliation for quantum cryptography**

Author(s): W. T. Buttler, S. K. Lamoreaux, J. R. Torgerson, G. H. Nickel and C. G. Peterson

Submitted to:

http://lib-www.lanl.gov/cgi-bin/getfile?00796756.pdf

Fast, efficient error reconciliation for quantum cryptography

W. T. Buttler, S. K. Lamoreaux, J. R. Torgerson, G. H. Nickel and C. G. Peterson
University of California, Los Alamos National Laboratory, Los Alamos, New Mexico 87545
(April 1, 2002)

We describe a new error reconciliation protocol *Winnow* based on the exchange of parity and Hamming's "syndrome" for $N-$bit subunits of a large data set. *Winnow* was developed in the context of quantum key distribution (QKD) and offers significant advantages and net higher efficiency compared to other widely used protocols (e.g. CASCADE). A detailed mathematical analysis of Winnow is presented as well as a comparison to CASCADE in the context of practical implementations of QKD; in particular, the information overhead required for secure implementation is one of the most important criteria in the evaluation of a particular error reconciliation protocol. The increase in efficiency for Winnow is due largely to the reduction in authenticated public communication required for its implementation compared to CASCADE.

PACS Numbers: 03.67.Dd, 03.67.Hk

I. INTRODUCTION

Quantum cryptography [1] presents special problems in regard to error correction of noisy quantum communications. Under the constraint that the public channel can be authenticated, and the assumption that all public communications can be eavesdropped, classical information on the exchanged qubits must be revealed through a series of public discussions to test the quantum key integrity and to remove the errors. Discrepancies within the qubits, observed as errors, must be treated as having been introduced by a hostile eavesdropper; the eavesdropper is generally referred to as Eve and labeled **E** in this work.

In a classical environment *all* errors can *always* be removed with the condition that to remove all errors one may have to reveal all information. However, within the secrecy framework imposed by quantum key distribution (QKD), revealed information reduces privacy. Because of this great care must be taken to reveal a minimal amount of information to remove errors from quantum key while accounting for the leaked information to ensure key integrity after errors are removed.

Within this context of QKD, the two parties that exchange qubits over a quantum channel (Alice (**A**) and Bob (**B**) is the notation typically used within the quantum cryptography community) must have a fast and efficient method to mend the quantum key; in addition, they must also reduce **E**'s knowledge gained during public discussions to a vanishingly small amount. These constraints require that any error reconciliation protocol will also need supporting protocols to provide a complete framework for quantum cryptographic security. That is, a useable QKD system will comprise a quantum-key transmitter (**A**) and receiver (**B**), and a series of protocols to remove errors and account for and mitigate the information leakage attributable to **E**. The series of protocols includes [2,3], but is not necessarily limited to the following: error-reconciliation [4,5], privacy amplification [6] and signature authentication [7].

In addition to these protocols, we acknowledge a protocol generally formulated in [4] that we refer to as privacy maintenance. We also note that the predecessor to CASCADE [5] — the best known and probably the most widely used error reconciliation protocol — is also generally formulated in [4]. The key difference between CASCADE and its predecessor is that CASCADE neglects privacy maintenance: all data are retained until the necessary privacy amplification is performed on the error-free data. We observe that the reconciliation process is much more efficient if privacy maintenance is implememted during reconciliation as will become obvious in the following discussion.

Finally, this work introduces a new error reconciliation protocol that uses a Hamming code [8,9] to remove errors. We refer to this protocol as *Winnow*. *Winnow* is characterized by the application of a parity test, a conditional Hamming hash, and privacy maintenance. The *Winnow* process describes freeing from the good bits the bad bits and is accurately analogous to the contemporary definition of *winnow*: to free (grain) from the chaff by fanning or forced air [10].

II. HAMMING ERROR DETECTION AND CORRECTION

A. Application of the Hamming Algorithm

The application of the Hamming hash function for error correction is illustrated as follows [8,9]:

First, after **A** and **B** exchange qubits on the quantum channel, **A** and **B** then divide their random bits into blocks of length $N_h = 2^m - 1$. (Due to the 1:1 correlation of these data, we henceforth refer to these blocks as a single data- or bit-block.) The $m-$bit ($m \geq 3$) *syndromes* S_a and S_b are then calculated, where S_a and S_b

resepectively depend only on **A**'s or **B**'s bits in a particular block.

Next, **B** transmits his syndrome to **A** and errors are only discovered if the syndrome difference S_d (*exclusive or* of S_a with S_b) is non-zero:

$$S_d = S_a \oplus S_b \neq \{0\}^m. \tag{1}$$

Finally, m bits are deleted from each bit block to eliminate the potential loss of privacy to **E** due to the (classical) communication of **B**'s syndromes: m bits of information are revealed on each block for which S_b is revealed. Specifically, data privacy is maintained by removal of m bits from each block at the $\{2^j\}$ positions where $0 \leq j \leq m-1$. These bits are independent in the syndrome calculations as shown in Eq. 4. We refer to the operation of discarding bits in this manner [4] as *privacy maintenance*.

The Hamming algorithm always corrects any single error within any bit block. The effect of the Hamming algorithm and privacy maintenance is less clear in the event that more than one error exists in a bit block. Such considerations are now discussed in detail.

1. Syndromes

The syndromes S_a and S_b are formed by contraction of the N_h-bit blocks with the matrix $h^{(m)}$:

$$S = \sum_{i=1}^{m} 2^i \cdot \mathrm{mod}\left(\sum_{j=1}^{N_h} X_j h_{i,j}^{(m)},\, 2\right), \tag{2}$$

where X_j represents bit j in **A**'s or **B**'s block.

The matrix $h^{(m)}$ is a special form of hash function [11] and is given by

$$h_{i,j}^{(m)} = \mathrm{mod}\left(\frac{j}{2^{i-1}},\, 2\right), \tag{3}$$

where $1 \leq i \leq m$, $1 \leq j \leq N_h$, and integer arithmetic is assumed (fractions are truncated). For example,

$$h^{(3)} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{4}$$

If a bit block contains zero errors, or if all of the bits within the block are in error (as can be verified by symmetry), $S_d = \{0\}^m$. If a block contains *exactly* one or two errors, or if all but *exactly* one or two bits of the block are in error, $S_d \neq \{0\}^m$. We can state this equivalently as $P(S_a \neq S_b) = 1$, for the number of bit errors $n_i \in \{1, 2, N_h-2, N_h-1\}$ where $P(S_a \neq S_b)$ is the probability that $S_a \neq S_b$ and n_i is the number of errors. Moreover, if a block contains *exactly* one bit error, S_d represents an m-bit binary number that indicates the bit-error position within the block. Unfortunately, the value of S_d does not indicate the location of multiple bit errors.

If there are more than two errors, and less than $N_h - 2$ errors within a bit block, S_d is evenly distributed among the 2^m possible syndromes. Of particular interest is the probability that $S_a = S_b \equiv P(S_a = S_b) = 1/2^m$ for $n_i \in \{3, 4, \ldots, N_h - 4, N_h - 3\}$.

In conclusion, note that more errors are introduced than are corrected, on average, for the cases which have $n_i \in \{2, 3, \ldots, N_h/2\}$.

B. Addition of Parity

Ideally, we would like to correct all errors in each bit block, introduce no additional bit errors, and reveal a minimal amount of information on the key bits to an eavesdropper through public communication.

The Hamming protocol outlined in the previous section has a number of shortcomings regarding this ideal. First, the difference syndrome S_d does not distinguish between single- and multiple-bit errors. Therefore, additional errors may be introduced if instances of $S_d \neq \{0\}^m$ are treated as due to single errors. Second, up to m bits of information are exchanged for each data block; information which can be compromised by eavesdropping.

One solution is to eliminate all bits within data blocks for which $S_d \neq \{0\}^m$. This certainly removes the possibility of introducing additional bit errors into the key, but, unfortunately, the efficiency of such a method is low as *every* block loses either m-bits to privacy maintenance, or all bits because $S_d \neq \{0\}^m$. The efficiency of this approach is not optimal as most of the discarded bits/blocks for which $S_d \neq \{0\}^m$ are probably not in error.

Another, much better solution is to introduce a preliminary parity comparison on a block of $N = 2^m$ bits and to make a comparison of the syndromes S_a and S_b conditional upon the result of the parity comparison.*

If the block parities do not agree an odd number of errors exists in the block. Moreover, if the bit errors

---

*Hamming discusses the addition of a parity check on the $N_h = 2^m - 1$ bit block [9] (pp. 47-48; pp. 213-214). His conclusion is that **A** and **B** are more likely to introduce additional errors than correct errors by changing a bit if $S_d \neq \{0\}^m$ and the block-parities agree. In this situation **A** and **B** could either remove the $m+1$ bits required to ensure privacy on the remaining bits (which may remove errors), or they could eliminate all of the bits in question, as $n_i \in \{2, 4, \ldots, 2^m - 2\} > 1$. The expanded protocol described in this effort allows the detection of an even or odd number of errors and prevents a correction attempt on those data blocks with even numbers of errors. This is important since the Hamming algorithm will increase the number of errors in blocks which have $n_i \in \{2, 3, \ldots, N_h/2\}$.

are distributed randomly throughout the data, and if the number of errors is sufficiently small, then an odd number of errors in a block probably indicates a single error which can be corrected by application of the Hamming algorithm. Therefore, in these instances, one bit is discarded for privacy maintenance and the Hamming algorithm is applied to the remaining N_h bits as previously discussed, and then $\lceil \log 2(N_h) \rceil$ additional bits are discarded to complete the privacy maintenance.

If the block parities agree, the syndromes are not calculated and compared, but one bit is still discarded from the block for privacy maintenance. We refer to this error reconciliation protocol as *Winnow*.

Winnow reveals no more than $\log_2(N)+1$ bits in 2 classical communications if the parities on the N bits do not agree: m bits for the syndrome and 1 bit for parity, whereas only 1 bit of information is revealed in 1 classical communication if the parities agree.[†]

Therefore, the amount of key data discarded is

$$N_{dis.}^{odd} = log_2(N) + 1 = m + 1 \tag{5}$$

bits for blocks with odd numbers of errors such that the fraction of the bits remaining after privacy maintenance is

$$\mu_{pm}^{odd} = 1 - \frac{N_{dis.}^{odd}}{N} \tag{6}$$

for blocks with odd numbers of errors. For $N \in \{8, 16, 32, 64, 128\}$, $\mu_{pm}^{odd} \in \{0.5, 0.69, 0.88, 0.89, 0.94\}$, respectively. Also,

$$\mu_{pm}^{even} = 1 - \frac{1}{N}, \tag{7}$$

and $\mu_{pm}^{even} \in \{0.88, 0.94, 0.97, 0.98, 0.99\}$ for the same values of N. In either case, the appropriate overhead for the classical communications is also removed immediately from the data so that the privacy of the bits is at least maintained if not improved.

III. REMOVING AND CORRECTING ERRORS

All single bit errors in a block are guaranteed to be either eliminated or corrected after a single pass of *Winnow* (a *Winnowing*). What remains to be considered is how blocks with multiple errors are affected.

[†]Exchanging the parity on $N = 2^m$ bits instead of $N_h = 2^m - 1$ bits results in slightly higher efficiency. That is: more information is revealed when the syndrome information is combined with the parity information on a N_h bit blocks than is revealed when the parity and syndrome are revealed on N bits in *Winnow*.

Define the change in number of errors in a given block and *for a given initial number of errors* as $\Delta n = n_f - n_i$, where n_i and $n_f \equiv n_f(n_i, N)$ are the initial and final numbers of bit errors in a block prior to and after *Winnowing*, respectively. The average change in the number of errors, for a given number of initial errors, after a *Winnowing* (this step includes elimination of the parity bit but not the final m-bits required for completion of the privacy maintenance step) can be expressed as

$$\bar{\Delta n} \equiv \langle \Delta n(n_i) \rangle = \sum_{\Delta n=-2}^{1} \Delta n \cdot p(\Delta n | n_i), \tag{8}$$

where

$$\sum_{\Delta n=-2}^{1} p(\Delta n | n_i) = 1, \tag{9}$$

and $p(\Delta n|n_i)$ is the probability that the number of errors will change by $\Delta n \in \{-2, -1, 0, 1\}$ given an initial condition of n_i errors in an $N-$bit data block. The $p(\Delta n|n_i)$ of interest can be written more instructively as

$$\begin{aligned}
p(+1|n_i) &= \pi^{(n)} \cdot \Pi_{S_d \neq 0}(n_i) \cdot \Pi^{(+)}(n_i) \\
p(\pm 0|n_i) &= \pi^{(n)} \cdot \Pi_{S_d = 0}(n_i) + \pi^{(y)} \cdot \Pi_{S_d \neq 0}(\delta n_i) \cdot \Pi^{(+)}(\delta n_i) \\
p(-1|n_i) &= \pi^{(n)} \cdot \Pi_{S_d \neq 0}(n_i) \cdot \Pi^{(-)}(n_i) + \pi^{(y)} \cdot \Pi_{S_d = 0}(\delta n_i) \\
p(-2|n_i) &= \pi^{(y)} \cdot \Pi_{S_d \neq 0}(\delta n_i) \cdot \Pi^{(-)}(\delta n_i),
\end{aligned} \tag{10}$$

where, n_i is as previously defined, $\delta n_i \equiv n_i - 1$, $\pi^{(y \vee n)}$ depends only on the initial number of errors (n_i) in the N-bit block and is the probability the bit discarded for privacy maintenance following the parity check was (y), or $(\vee)$ was not (n) in error; $\Pi_{S_d=0}(n_i^{\vee} \delta n_i)$ and $\Pi_{S_d \neq 0}(n_i^{\vee} \delta n_i)$ are the probabilities that $S_a = S_b$ or $S_a \neq S_b$ for n_i or δn_i errors in N_h bits, and $\Pi^{(\pm)}(n_i^{\vee} \delta n_i)$ is the probability that the number of errors $(n_i^{\vee} \delta n_i)$ in the N_h bits changes by $\Delta n = \pm 1$ following the prescription of the Hamming hash for instances in which $S_a \neq S_b$.

Eq. 8 can be expressed in terms of $\{\pi^{(y \vee n)}, \Pi_{S_d}, \Pi^{(\pm)}\}$ as

$$\begin{aligned}
\bar{\Delta n} &\equiv \langle \Delta n^{(n)}(n_i) \rangle + \langle \Delta n^{(y)}(n_i) \rangle \\
&= \bar{\Delta n}^{(n)} + \bar{\Delta n}^{(y)},
\end{aligned} \tag{11}$$

where the arguements which depend on n_i have been supressed, and

$$\begin{aligned}
\bar{\Delta n}^{(n)} &= \pi^{(n)} \cdot \Pi_{S_d \neq 0}(n_i) \cdot \Big[1 - 2 \cdot \Pi^{(-)}(n_i) \Big], \\
\bar{\Delta n}^{(y)} &= \pi^{(y)} \cdot \Pi_{S_d \neq 0}(\delta n_i) \cdot \Big[1 - 2 \cdot \Pi^{(-)}(\delta n_i) \Big] - \pi^{(y)}.
\end{aligned} \tag{12}$$

From the $\{\Pi_{S_d}, \Pi^{(\pm)}\}$ shown in Table I and the equalities from Eq. 13, the $\{\pi^{(y \vee n)}, \Pi_{S_d}, \Pi^{(\pm)}\}$ of interest for *Winnow* can be calculated:

$$\begin{aligned} \pi^{(y)} &= \frac{n_i}{N} \\ \pi^{(y)} + \pi^{(n)} &= 1 \\ \Pi_{S_d \neq 0}(n_i^{\vee}\delta n_i) + \Pi_{S_d=0}(n_i^{\vee}\delta n_i) &= 1 \\ \Pi^{(-)}(n_i^{\vee}\delta n_i) + \Pi^{(+)}(n_i^{\vee}\delta n_i) &= 1. \end{aligned} \tag{13}$$

In Table II we introduce a new quantity

$$\bar{n}_f \equiv \langle n_f \rangle = n_i + \bar{\Delta n}, \tag{14}$$

and in Table III we define a new parameter

$$p_f = \frac{\bar{n}_f}{N_f}. \tag{15}$$

The parameter p_f defines the probability for each bit in a given block to be in error. The number $N_f \in \{N-1, N-m-1\}$ and its value depends on the action required by *Winnow* for a given number of initial errors. For example, $N_f = N-1$ or $N-m-1$ for p_f and n_i even or odd, respectively.

These two tables show the effect of *Winnow* on data which are divided into 8-bit blocks. The values marked with superscript p reflect the effect of discarding one bit following the parity comparison. The values marked with superscript *ph* refer to the data after the Hamming algorithm is also applied, but before the requisite $\log_2(N) = 3$ bits of data are discarded for privacy maintenance. The final values denoted by subscript f reveal the effect of *Winnow* (including the effect of all discarded data required for privacy maintenance).

The parameter p_f clearly shows a reduction in errors for $n_i = 1$ and an increase in errors for $n_i = 3$. It also shows that discarding data to maintain privacy of the remaining key has no effect on the error probability.

A. Probability for Residual Errors

The fraction of key remaining after a *Winnowing* is given by

$$\mu_N \equiv \frac{\langle N_f \rangle}{N} = \frac{\sum_{n_i=0}^{N} N_f \, P(n_i|N)}{N}, \tag{16}$$

and the probability for any key bit to be in error following a *Winnowing* is

$$p_N = \frac{\langle \bar{n}_f \rangle}{\langle N_f \rangle} = \frac{\sum_{n_i=0}^{N} \bar{n}_f(n_i) \cdot P(n_i|N)}{N \cdot \mu_N}, \tag{17}$$

where $P(n_i|N)$ is the probability for an N-bit block to contain n_i errors before a *Winnowing*.

1. Random Distribution of Errors

Obviously, the efficiency with which *Winnow* removes errors depends upon the distribution of errors within the data. Without intimate knowledge of a specific QKD apparatus, a reasonable assumption is that the errors are random and normally distributed throughout the data. Given this assumption, $P(n_i|N)$ in Eq. 17 is given by the binomial distribution

$$P(n_i \mid N, p_0) = \binom{N}{n_i} {p_0}^{n_i} (1-p_0)^{N-n_i} \tag{18}$$

where p_0 is the probability that any given bit is in (relative) error.

With this assumption, Eqs. 16 and 17 can be expressed as

$$\mu_N = \frac{N - 1 - m \sum_{n_i^{odd}} \binom{N}{n_i} {p_0}^{n_i} (1-p_0)^{N-n_i}}{N}, \tag{19}$$

where $m = \log_2(N)$, and

$$p_N = \frac{\sum_{n_i=0}^{N} \bar{n}_f(n_i) \binom{N}{n_i} {p_0}^{n_i} (1-p_0)^{N-n_i}}{N \cdot \mu_N}. \tag{20}$$

IV. ANALYSIS

The efficiency with which *Winnow* reduces errors in the key is of great interest. Two related issues which concern the efficiency are: 1) the number of iterations of *Winnow* necessary to achieve a sufficiently low probability of error in the remaining key data, and 2) the amount of key data that is discarded through privacy maintenance.

The number of iterations is of concern because each iteration reveals information and consumes time with each communication between **A** and **B**. Moreover, each communication requires the use of some private key for signature authentication [7]. Most importantly, though, is that each iteration requires a significant amount of data to be discarded through privacy maintenance.

Smaller N require more data to be discarded than larger N as can be seen from Eq. 19. However, an effect which tends to mollify this undesirable condition is that smaller N are more efficient at removing errors for larger values of initial error probability. This effect is illustrated in Fig. 1 where we have plotted p_N/p_0 for several values of N. For all values of N and p_0 sufficiently small, $p_N/p_0 < 1$ and the protocol can remove errors from the key data. However, as p_0 increases from $p_0 = 0$, each of the curves passes through $p_N/p_0 = 1$ indicating that additional errors are being introduced into the key. Moreover, the value of p_0 for which $p_N/p_0 = 1$ is smaller for larger N and the curves do not intersect between $p_0 = 0$ and $p_N/p_0 = 1$.

A. The Iterated Application

As a primary requirement of *Winnowing* real data in an iterative application, a random shuffling of the data between iterations is essential to randomly redistribute missed or introduced errors. Without this random shuffle multliple errors remain clumped together and, in essence, are impossible to completely remove from the data. Under this constraint it is obvious that the final error probability, and the amount of data remaining after a number of *Winnowings*, depends on the way in which N is varied throughout the successive *Winnowings*. An intuitive result which we have verified empirically is that less data are discarded for the same initial and final error probabilities if N is chosen well for the first iteration and is either held constant or increased for all subsequent iterations; there is no advantage to decreasing N in subsequent iterations if *Winnow* is applied as outlined here.

Define

$$p(p_0; \{j_N\}) \tag{21}$$

and

$$\mu(p_0; \{j_N\}) \tag{22}$$

as the final error rate and fraction of data remaining after a sequence $\{j_N\} = \{j_8, j_{16}, j_{32}, j_{64}, j_{128}\}$ where j_N iterations of *Winnow* are applied with a block size $N \in \{8, 16, 32, 64, 128\}$ beginning with $N = 8$ and increasing monotonically in N by factors of 2.

In this work N is constrained such that $N \leq 128$ only for the sake of brevity. We have found that this constraint does not impose a serious limit on the ability of *Winnow* to correct errors. The ideas discussed below can be extended to include $N > 128$ in a straightforward manner.

Because $p_8 < p_0 \; \forall \; p_0 < 0.5$, it may appear that errors can be corrected in the data for this entire range of initial error probability. However, there is another criterion that must be met which significantly reduces the maximum correctable error probability: There must remain a finite amount of error-free data after the potential information possessed by **E** is reduced through privacy amplification.

B. Eavesdroppers and the BB84 scheme

The maximum amount of potential information possessed by **E** can be determined by the initial error probability p_0 and depends on the QKD protocol and the type of attacks being employed. For example, if the BB84 protocol is used and **E** employs a complete intercept/resend attack on the quantum channel in the same bases used by **B**, she will introduce an error probability of $p_0 = 1/4$. She will also potentially know $1/2$ of the data before error reconciliation and up to $2/3$ of the data which remains after error reconciliation.

If **E** uses a more clever intercept/resend strategy of detecting and resending in the Breidbart basis (second paper in [4]), she would introduce the same number of errors ($p_0 = 1/4$) and could know up to a fraction of 0.59 of the data before error reconciliation and 0.78 of the data remaining after error reconciliation.

It should also be noted that certain states of light are more susceptible to attack than others. For example, consider weak coherent states which are commonly used in QKD systems. If **E** also employs a beamsplitter attack [3,4,12] to one of these systems, an additional amount of data is compromised which is not greater than the mean number of photons in the state. However, this value can be made arbitrarily small so it is neglected in the following calculations. Moreover, other states of light can be used in QKD schemes which are not vulnerable to this type of attack [13].

Thus, the fraction of data remaining after error reconciliation and privacy amplifications can be

$$\nu^{bb84} = \mu - (0.59) 4\, p_0 \tag{23}$$

for BB84, where ν describes the remaining fraction of key.

C. Reconcilable Errors

From the above considerations, p and ν can be investigated as a function of p_0. Of particular interest is the maximum p_0 for which some secure data remains while achieving a sufficiently low final error probability to make the data useful. We have chosen, somewhat arbitrarily, $p \leq 10^{-6}$ as a reasonable target for the final error probability.

With this target and the remaining fraction of private data described by Eq. 23, we find the largest initial error probability for which some private data remains is

$$p_0 = 0.135, \tag{24}$$

after *Winnowing* and privacy amplification.

To achieve $p \lesssim 10^{-6}$ from this large initial error probability, *Winnow* must be applied in the sequence $\{j_N\} = \{3, 0, 1, 2, 2\}$. That is, 3 *Winnowings* with $N = 8$ must be followed by 1 *Winnowing* with $N = 32$, *etc.* If this prescription is followed,

$$\nu^{bb84} = 0.002 \tag{25}$$

of the original data remain and are secure following privacy amplification.

Some QKD schemes require a larger estimate of **E**'s knowledge. If Eq. 23 is replaced with [4]

$$\nu = \mu - 2\sqrt{2}\, p_0, \tag{26}$$

we find

$$p_0 = 0.123 \tag{27}$$

for $\{j_N\} = \{2, 1, 1, 1, 4\}$. This leaves a fraction $\nu = 0.005$ of the original data as secure data with a single-bit error probability $\leq 10^{-6}$.

Finally, if we estimate that **E** knows every bit of data by causing $p_0 = 1/4$, then

$$\nu = \mu - 4\, p_0. \tag{28}$$

We then find that the largest reconcilable p_0 is

$$p_0 = 0.104 \tag{29}$$

for $\{j_N\} = \{2, 0, 2, 0, 4\}$ and $\nu = 0.004$.

The most efficient iteration sequence ($\{j_N\}$) for any QKD scheme can be determined by first applying *Winnow* with $N = 8$ to estimate p_0. Once the number of blocks with odd and even (even includes zero) errors, M_e^{odd} and M_e^{even} respectively, are known, the fraction

$$\frac{\#\text{ of Parity Errors}}{\#\text{ of Blocks}} = \frac{\sum_{n_i^{odd}} \binom{N}{n_i} {p_0}^{n_i} (1 - p_0)^{N - n_i}}{N} \tag{30}$$

can be used to estimate p_0. Knowledge of p_0 is sufficient to determine the $\{j_N\}$ which maximizes ν.

For small p_0, the most efficient $\{j_N\}$ may start with $N > 8$. However, currently working systems which have been reported in the literature have large enough error probabilities so that the most key is left if $N = 8$ for at least the first iteration.

V. COMPARISON WITH CASCADE

A detailed analysis of the advantages of *Winnow* over other protocols is beyond the scope of this work. However, it is instructive to note the advantages over at least the best-known protocol CASCADE.

The most notable difference between *Winnow* and CASCADE is that CASCADE does not employ privacy maintenance. The disadvantage of such a protocol is that super-redundant information must be exchanged with each successive iteration. This is to be compared with CASCADE's predecessor and *Winnow* which reduce the size of the data set with each communication. With the reasonable requirement that a bit revealed through these communications requires at least a bit to be eliminated through some channel, either before or during privacy amplification, then the inefficiency of CASCADE becomes obvious: retaining and repetitively exchanging information on the same bits is an additional expense to the protocol.

For the purpose of comparison, we have computed the maximum p_0 which CASCADE can successfully reconcile errors and preserve a small amount of secure data after privacy amplification and the removal of the super-redundant information. We find

$$p_0 = 0.114 \tag{31}$$

for $\{j_N\} = \{2, 1, 0, 2, 1\}$ and $\nu^{bb84} = 0.01$ when $(0.59)4\, p_0$ describes the additional amount of key that must be discarded through privacy amplification. This is to be compared with $p_0 = 0.135$ for the same considerations with *Winnow*. Obviously *Winnow* can claim an advantage over CASCADE just on these grounds.

However, this comparison (or any of the previous *Winnow* discussion) does not take into account bits used to authenticate messages sent between **A** and **B**. CASCADE requires significantly more two-way communication than *Winnow*, and each packet of n bits sent may require $\lceil \log_2 n \rceil$ for authentication [7]. We calculate that the most efficient application of CASCADE requires $1 + \log_2 N$ communications per iteration while *Winnow* requires only 2 communications *for any block size* N that exhibits a parity error. CASCADE can be made somewhat more efficient by including privacy maintenance, but the additional communication required imposes a tight limitation on its practical efficiency. In addition, because CASCADE does not maintain privacy, subsequent iterations requires more bits to be exchanged in the initial parity phase with each iteration. The additional bit exchanges require additional signature authentication bits.

In the spirit of full disclosure we unequivicolly state that because CASCADE and its predecessor always removes a single error and never introduces additional errors to multiple error blocks, both CASCADE and its predecessor perform infinitesimally better than *Winnow* in an environment where signature authentication is not required and privacy maintenance is removed from the *Winnow* protocol. However, this miniscule efficiency improvement in non-authenticated and non privacy enhanced data does not offset the time lost by CASCADE and its predecessor due to the many additional communications, i.e. *Winnow's* 2 communications is a great advantage where time is of the essence with regard to production of such a precious commodity as private, secure key bits over inefficient noisy quantum channels. Consider that while CASCADE and its predecesor are still negotiating to reconcile key that *Winnow* is building more key. When signature authentication is factored in with the additional communications *Winnow's* superiority cannot be denied.

VI. DISCUSSION

We have empirically verified that *Winnow* distills error-error free key bits at a much faster rate than CASCADE or its predecessor on a functional QKD system

[14]. The empirical results verify that CASCADE's predecessor is infinitesimally more efficient, but much slower at distilling error free key. Regarding CASCADE, it has also been empirically verified that *Winnow* and CASCADE's predecessor are much more efficient with higher error rates. The mathematics formalized herein support the empirical results.

VII. CONCLUSION

We have identified a new, fast, efficient, error reconciliation protocol for quantum key distribution which requires only 2 communications between the two parties attempting to generate private key material.

This new protocol (*Winnow*) incorporates a preliminary parity comparison on blocks whose size is $N = 2^m$ where $m \in \{3, 4, 5, 6, ...\}$. Subsequently, one bit is discarded from these blocks to maintain the privacy of the remaining bits. A Hamming hash function, which can be used to correct single errors, is applied to the remaining $N-1$ bits on the blocks whose parities did not agree. Finally, m bits are discarded from the blocks on which the Hamming algorithm was applied to maintain the privacy of those bits.

We find this protocol capable of correcting an initial error probability of up to 13.5% in privacy amplified BB84-like quantum key distribution schemes, which is to be compared with CASCADE which can correct error rates up to 11.4% in similar systems.

Acknowledgements: The authors extend their thanks and appreciation to R. J. Hughes, E. Twyffort, D. P. Simpson and J. S. Reeve for many helpful discussions regarding this effort.

[1] C. H. Bennett and G. Brassard, "Quantum cryptography, public key distribution and coin tossing," *International Conference on Computers, Systems & Signal Processing, Bangalore, India, 1984* (IEEE, New York, 1984) 175-179; A. K. Ekert, "Quantum cryptography based on Bell's theorem," Phys. Rev. Lett. **67**, 661-663 (1991); C. H. Bennett and S. J. Wiesner, "Communication via one- and two-particle operators on Einstein-Podolsky-Rosen states," Phys. Rev. Lett. **69**, 2881-2884 (1992); C. H. Bennett, "Quantum cryptography using any two non-orthogonal states," Phys. Rev. Lett **68**, 3121-3124 (1992).

[2] C. Cachin and U. M. Maurer, "Linking information reconciliation and privacy amplification,"J. Cryptology **10**, 97-110 (1997).

[3] N. Lütkenhaus, "Estimates for practical quantum cryptography," Phys. Rev. A **59**, 3301-3319 (1999).

[4] C. H. Bennett, F. Bessette, G. Brassard, L. Salvail and J. Smolin, "Experimental Quantum Cryptography," Lect. Notes in Comput. Sci. **473**, 253-265 (1990); J. of Cryptology **5**, 3-28 (1992).

[5] G. Brassard and L. Salvail, "Secret Key Agreement by Public Discussion," Lect. Notes Comput. Sci. **765**, 410-423 (1994).

[6] C. H. Bennett, G. Brassard, C. Crepeau and U. M. Maurer, Generalized Privacy Amplification," IEEE Trans. Inf. Theory **41**, 1915-1923 (1995).

[7] W. Diffe and M. E. Hellman, "Multi-user cryptographic techniques," Proceedings of AFIPS National Computer Conference, 109-112 (1976); R. L. Rivest, A. Shamir and L. M. Adleman, "A method for obtaining digital-signatures and public-key cryptosystems," Communications of the ACM **21**, 120-126 (1978); C. Mitchell, F. Piper and P. Wild, "Digital Signatures," G. J. Simmons (Ed.), *Contemporary Cryptography: The Science of Information Integrity*, 325-378 IEEE Press, 1992.

[8] R. W. Hamming, "Error detecting and error correcting codes," The Bell Syst. Technical. Journ. **2**, 147-161 (1950).

[9] R. W. Hamming, *Coding and Information Theory*, Prentice Hall, 239 pp, New Jersey (1980).

[10] *Random House Webster's College Dictionary*, Copyright 1992, 1991, by Random House, Inc., New York, NY.

[11] A. J. Menezes, P. C. van Oorschot and S. A. Vanstone, *Handbook of Applied Cryptography*, 780 pp, CRC Press, New York (1997).

[12] M. Dušek, O. Haderka and M. Hendrych, "Generalized beam-splitting attack in quantum cryptography with dim coherent states," Optics Communications **169**, 103-108 (1999).

[13] J. Kim, O. Benson, H. Kan and Y. Yamamoto, Nature **397** 500-503 (1999); P. Michler et al., Science **290**, 2282 (2000); B. Lounis and W. E. Moerner, Nature **407**, 491 (2000); Note: of the several single photon sources that have been proposed and demonstrated none of them are at a useful stage of development at this time.

[14] W. T. Buttler, R. J. Hughes, S. K. Lamoreaux, G. L. Morgan, J. E. Nordholt, and C. G. Peterson, "Daylight quantum key distribution over 1.6 km," Phys. Rev. Lett. **84**, 5652-5655 (2000); R. J. Hughes, G. L. Morgan and C. Glen Peterson, "Quantum key distribution over a 48 km optical fibre network," J. Modern Optics **47**, 533-547 (2000).

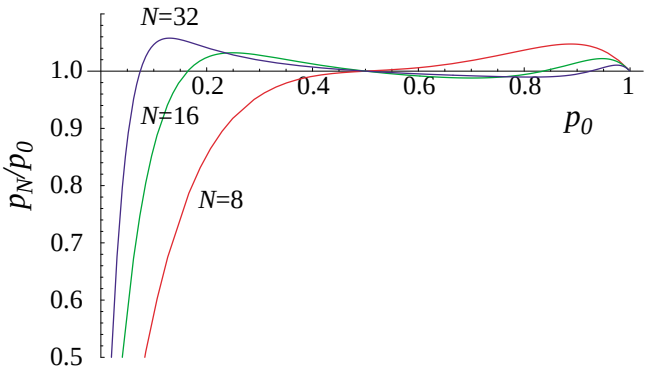

FIG. 1. The ratio p_N/p_0 for $N = 8$, 16 and 32. These curves illustrate the change in the probability that a given bit is in error after a single application of *Winnow* for the indicated block size N. Note that $p_8 < p_{16} \; \forall \; p_0 < 0.5$. Also note that $p_{16} < p_{32} \; \forall \; p_0 < 0.17$. This indicates that applications of *Winnow* with smaller N are more efficient at removing errors than are applications with larger N.

TABLE I. Summary of probabilities $\{\pi^{(y \vee n)}, \Pi_{S_d}, \Pi^{(\pm)}\}$.

$n_i/block$	1	3	...	$N-3$	$N-1$
$\Pi_{S_d \neq 0}(n_i)$	1	$1-\frac{1}{N}$	$1-\frac{1}{N}$	1	0
$\Pi_{S_d \neq 0}(\delta n_i)$	0	1	$1-\frac{1}{N}$	$1-\frac{1}{N}$	1
$\Pi^{(-)}(n_i)$	1	$\frac{3}{N-1}$	$\frac{n_i}{N-1}$	$\frac{N-3}{N-1}$	$1^{\dagger}$
$\Pi^{(-)}(\delta n_i)$	0	$\frac{2}{N-1}$	$\frac{n_i-1}{N-1}$	$\frac{N-4}{N-1}$	0

† This case (and $\Pi^{(+)}(0)$) is special as all (or no) bits are in error. The Hamming code prevents us from making a bit change that would decrease (increase) the number of errors in this instance.

TABLE II. $\bar{n}_f$ for $N = 8$ for various stages in *Winnow* (note that *Winnow* is not applied to blocks that contain an even number of errors).

n_i	0	1	2	3	4	5	6	7	8
$\bar{n}_f^p$	0	0.88	1.75	2.63	3.5	4.38	5.25	6.13	7
$\bar{n}_f^{ph}$	0	0	1.75	2.86	3.5	4.14	5.25	7	7
$\bar{n}_f$	0	0	1	1.64	2	2.36	3	4	4

TABLE III. $\bar{n}_f/N_f$ for $N = 8$ for various stages in *Winnow* (note that the Hamming component of *Winnow* is not applied to blocks that contain an even number of errors).

p_i	0	0.13	0.25	0.38	0.5	0.63	0.75	0.88	1
p_f^p	0	0.13	0.25	0.38	0.5	0.63	0.75	0.88	1
p_f^{ph}	0	0	0.25	0.41	0.5	0.59	0.75	1	1
p_f	0	0	0.25	0.41	0.5	0.59	0.75	1	1